

**ments Formula APL/S**

**e**  $R^2\pi$   $R*2\pi$

$1.10 B_i$   $B=100\ 35\ 200\ 75$   
 $1.10xB$

amounts. Compute inflated amounts.)

**of**  $\sum P_i S_i$   $P=53.75\ 220.50\ 87$   
 $S=100\ 50\ 75$

for each stock.  
for each stock. Compute market value.)

**ues**  $\sum x_i$   $XI=10\ 20\ 30\ \dots\ 100$

le  $XI$ . Sum all elements.)  $+/XI$

$\frac{\sum x_i}{N}$   $N=SIZE\ XI$   
 $+/XI\div N$

size of  $XI$ . Compute mean.)

**ation**  $\sqrt{\frac{\sum (x_i - \mu)^2}{N}}$   $M=+/XI\div N$   
 $+/(XI-M)*2\div N*.5$

d store mean. Compute standard deviation.)

rise to power of)

**ince**  $\frac{1}{RT} = \sum \frac{1}{R_i}$   $R=100\ 500\ 1000$   
 $\div +/R$

nce values. Compute answer.)

**an**  $N\sqrt{IIR_i}$   $R=1.4\ 1.7\ .9$   
 $x/RI*\div SIZE\ RI$

nts. Compute mean.)

**ilue**  $\sum \frac{CF_i}{(1+R)_i}$   $CF=1000\ 2000\ 2000$   
 $I=INDEX\ SIZE\ CF$   
 $+/(CF\div (1.10*I))$

ows. Generate arrays with values from 1 to size  $CF$ .  
 $R=.10$ .)

## chnical Specifications

alar, array and program names.  
es programs, scalars and arrays.  
work space on tape.  
s work space from tape.  
ifies tape can be loaded without disturbing  
rent workspace.  
edit mode.  
**nands**  
active backspace.  
es current line and allows entry of new line.  
iles current line and displays next line.  
Compiles current line and displays previous line.  
mpiles current line and allows insertion of new  
e.  
eturns to immediate mode.

- Transfers control to specified program.
- ula** — Assignment statement.
- uates and displays results.
- stant"** — Displays a string of characters.
- Controls bar graph heights.
- Controls bar graph colors.

**WHILE condition** }  
**DO block** } repetition block  
**ENDW** }

**PROG program-name block** }  
**ENDP** } program block

**ONTR** — Turns on tracing during program execution.  
**OFFT** — Turns off tracing during program execution.  
**KEYB** — Keyboard read function

### Two Argument Functions (value<sub>1</sub> function value<sub>2</sub>)

where value<sub>1</sub> and value<sub>2</sub> can be a scalar or array.

|     |                        |     |                    |
|-----|------------------------|-----|--------------------|
| +   | addition               | GT  | greater than       |
| —   | subtraction            | LE  | less than or equal |
| x   | multiplication         | LOG | any base logarithm |
| ÷   | division               | LT  | less than          |
| *   | power                  | MAX | maximum            |
| ,   | concatenation function | MIN | minimum            |
| AND | and                    | MOD | any base modulo    |
| EQ  | equal                  | NE  | not equal          |
| GE  | greater than or equal  | OR  | or                 |

### Reduction Function (function/value)

where function is any Two-Argument Function and value is an array.

Example:  $+/CF$  returns the sum of all values in the array  $CF$ .  
General:  $+/CF$  translates into a  $CF_1 + CF_2 + \dots + CF_n$  expression.

### One-Argument Functions (function value)

where value can be a scalar or array.

|       |                 |       |                         |
|-------|-----------------|-------|-------------------------|
| +     | unary plus      | FLOOR | lower integer           |
| —     | unary minus     | INDEX | index generator         |
| ÷     | reciprocal      | LOG   | natural log             |
| !     | factorial       | NOT   | not                     |
| ABS   | absolute value  | RAND  | random number generator |
| ARRAY | create an array |       |                         |
| CEIL  | higher integer  | SIGN  | sign of value           |
| COS   | cosine          | SIN   | sine                    |
| EXP   | expotential     |       |                         |

### Technical Notes

Values of variables can be either a scalar or an array.  
All values are stored in 4-byte floating point.  
Number Range is  $\pm 3.6 E \pm 75$ .  
Precision is 6 to 7 digits.  
E-notation input and output.

64 character text display in 4 physical lines.  
125 character maximum logical line.  
7 bars for graphics (16 colors).

1K bytes of RAM for TV support and internal buffers.

1K bytes of RAM for program and array storage.

- keywords are stored as single byte tokens.
- variable references are stored as single byte tokens.
- memory available for up to 1000 steps or 250 values.

The APL/S Language is available at a savings if you buy as part of VideoBrain System 400. System 400 includes the computer, Expander 1, APL/S, VB-81 (Financier) and VR-1000 (Money Minder).

**VideoBrain™**

# APL/S

## The Computational Language

The new programming language from VideoBrain for business and financial analysis, science and entertainment and education.

### 1. An Introduction to APL/S The Computational Language

APL/S was designed specifically to get the most out of a small computer. It takes some of the best features of BASIC, PASCAL, and APL and combines them to create a unique language with all of the following features:

- Array capability facilitates handling large groups of numbers clearly and concisely.
- Structured format makes it easy to write or change, and debug programs.
- Derivation from Iverson's mathematical systems means your concept of what the program should do leads to the program itself in a straightforward manner.
- 'Condensed' programs can do very complex, powerful tasks.
- Simplified Input/Output means no hangups getting data into or out of the computer.
- 'Immediate Mode' allows you to easily execute elaborate computations without even writing a formal program.
- Built-in color bar graph visually presents your output.
- Full floating point package with range  $\pm 10^{\pm 75}$

### 2. Putting APL/S to Work

There are thousands of uses for APL/S. A few of them are listed here:

#### Financial & Business Analysis

TAX PLANNING: Form 1040 Computation, etc.

REAL ESTATE: Depreciation Schedule, etc.

SECURITIES ANALYSIS: Ratio Analysis, etc.

OPTIONS ANALYSIS: Theoretical Value of a Call or Put, etc.



### Analysis Example: Stock Price Beta

#### Comments

Start of program named Beta.  
Get array of stock's prices over time from keyboard input.  
Get array of market index prices from keyboard.  
Compute changes

LINE SIZE M-1  
1)=P(I)  
I)=M(I)

MH)=+/(MHxMH) Compute Beta using reduction operators.  
Display Beta.  
End of Program

### Estimating Income Tax — Joint Return

#### Comments

Start of program named TAX.  
Get taxable amount from keyboard.  
Call program CTX to compute tax.  
Title output "TAX"  
Display value of Tax.  
End of Program.

I)=4000 Start of program named CTX  
Determine tax bracket  
80, 2260, 3260, 4380, 5660, 7100, 8660  
340, 12140, 14060 Tax Schedules  
, .25, .28, .32, .36, .39, .42,  
, .50  
+SLOP(I)x(TXA-(Ix4000+3200)) Compute Tax  
End of Program

### Engineering

PLS: Polynomial Evaluations, etc.  
CHEMISTRY: Orbits and Trajectories, etc.  
AERONAUTICS: Coordinates to Distance, etc.

### Example: A Generalized Polynomial Evaluator

$$F(x) = \sum_{i=0}^N C_i \cdot x^i$$

#### Comments

Start of program named Poly.  
Get array of constants from keyboard.  
Get array of x values from keyboard.  
Create an array of I values  
(1, 2, 3, ..., i-1, i)  
Compute and display result.  
End of Program.

### Recreation and Entertainment

PLS: Business, etc.  
MATHEMATICAL EXERCISES: Number Guessing, etc.  
GAMES: Caves, etc.

### Science Fiction Example: Earthlander Simulation

#### Program

PROG EARTH  
BAR COLOR 0 2 4 2 6 2 14  
V=0  
=6000  
=32  
FUEL=2000  
CLOCK=0  
WHILE D GT 0  
DO B=KEYB  
T=KEYB  
IF B GT 0  
THEN BT=FUEL÷B MIN T

ELSE BT=T  
ENDIF  
D=D-(VxT+T\*2xG÷2  
-BTx2xB÷2  
V=V+GxT-BxBT  
FUEL=FUEL-BTxB  
CLOC=CLOC+T  
BARHeight (D,0,V,0,  
FUEL,0,CLOCx100)  
=6000x64  
D,V,FUEL,CLOC

ENDWHILE  
IF V LT 15  
THEN "SUCCESS"  
ELSE "CRASH"  
ENDI  
V,CLOC,2000-FUEL

#### Comments

Start of program named EARTH  
Set of bar colors  
Velocity  
Distance to earth.  
Gravitational constant  
Available fuel  
Clock

Get burn rate  
Get burn duration.  
If burn rate is greater than zero  
set burn time according to  
available fuel  
Else set burn time to input value.

Update distance

Update velocity  
Update fuel gauge  
Update clock  
Set bars and scale them

Display distance, velocity, fuel  
and time  
If terminal velocity is less than  
15 ft/sec, then success—other-  
wise you crashed

Display terminal velocity, clock  
fuel used.

### 3. Learning to Program with APL/S

There are three criteria for selecting a programming language:

- It must be useful once you've learned it.
- The programming principles you learn must be applicable to other programming languages.

The language must be easy to learn and use and be backed up with perfectly thought-out error statements (to catch your errors while programming) and solid instruction and reference manuals. "Application programs" should be available to expand the usefulness of the language into areas you may not have thought of or haven't the time or expertise to write programs in yourself.

APL/S was designed to meet all of the above criteria better than any other small computer language.

APL was originally designed to facilitate conceptual thinking and problem solving with logical algorithms. It is frequently recognized as the most "elegant" and straightforward programming language. The programming lessons which are abundantly clear in APL/S will serve you well no matter what language you may add to your repertoire in the future.

APL/S is extremely easy to learn and use. And being essentially straightforward, its structured format makes it easy to tell what should be happening at any point in the program. This makes it easy to write programs, easy to detect conceptual errors, and easy to debug (change and correct errors). Your

Stanford University specialists in programming and education makes it easy to learn the language one step at a time. And the manual's complete reference section will make quick work of answering your questions.

### 4. APL/S vs. BASIC

Many of the programming statements and functions available in BASIC are very similar to those in APL/S. The advantages of APL/S stand out sharply, however, when you begin designing programs for useful purposes.

APL/S is more powerful than BASIC because the APL/S array capability can group many numbers and handle them clearly and concisely.

APL/S is full floating point (able to handle numbers in decimal form) ranging  $\pm 10^{\pm 75}$ .

APL/S has a structured format that makes it easier to write programs, debug them, and change them.

An APL/S program itself is similar to your own concept of the problem you want to solve. This means there's less chance of confusion and error. In BASIC it is easier to err because you have to translate your conceptual solution to the actual BASIC program. APL/S programs are generally simpler and shorter than BASIC programs too. The following programs are designed to take the number of shares of each stock held in a portfolio and the price of each stock to compute the total value of the portfolio. Example:

#### BASIC

```
10 READ N
20 MV=0
30 FOR I=1 to N
40 READ SHS,PRC
50 MV=SHSxPRC+MV
60 NEXT I
70 PRINT MV
100 DATA 3
110 DATA 100 53.75
120 DATA 50 220.50
130 DATA 75 87.00
RUN
22925
```

#### APL/S

```
PROG PORT
SHS = 100 50 75
PRC = 53.75 220.50 87
+/(SHSxPRC)
ENDP
PORT
22925
```

As you can see, APL/S is not just shorter — it's a lot easier to understand.

### 5. APL/S Immediate Mode

The immediate mode of APL/S allows you to use most of the language capabilities in an interactive way. Without even writing a formal program you can input data and formulas to get quick answers. You can even change parts of the data and formulas to get new answers without re-entering any of your previous inputs. The following table shows some examples of using APL/S in the Immediate Mode.